A Comparative Study of ARL Linux Cluster Performance

George Petit, James Ianni, and Martin Monaghan
US Army Research Laboratory (ARL)/Raytheon Company, Aberdeen Proving Ground, MD
{gpetit, james.ianni, mmonaghan}@arl.army.mil

Abstract

Since 2003, with the installation of a 256 processor Linux Networx XEON 1686 system (Powell), Army Research Laboratory (ARL) has been providing large-scale Linux cluster production systems for use within the Department of Defense (DoD) High Performance Computing Modernization Program (HPCMP). This initial system was followed in 2004 with the 2,048 processor Linux Networx XEON EM64T (JVN) and 2,304 processor IBM Opteron (Stryker) systems, and in 2007 by the 3,368-core Intel Dempsey (Humvee) and 4,488-core Intel Woodcrest (MJM) systems. These latest systems provide an increased peak performance of over 15 times the original Xeon 1686 system in a four year period. The purpose of this paper is provide a comparative study of the three generations of Linux clusters' capabilities to process some of the most widely used production codes used within the ARL environment. The codes to be benchmarked will include CTH, CFD++, GAMESS, and OVERFLOW. The codes will be run on Powell, JVN and Ping (the Woodcrest testbed machine). The results will focus attention on how architecture enhancements (including CPU speed, memory per node, cache size, and interconnect fabric transfer rates) have affected the overall performance of these systems.

1. Introduction

The purpose of this study was to compare the performance of the three generations of Linux clusters that have been installed at the ARL Major Shared Resource Center (MSRC) since 2003. Several of the most highly utilized software packages within the center were run on each machine with representative input files to simulate production performance. It was our intention at the outset of this project to utilize instrumented versions of each software package to obtain profile data to analyze the performance differences. Unfortunately, most of the instrumentation software we had hoped to use was not available across all three platforms, especially for the Woodcrest architecture. Therefore, most of the analysis is based on observations made from wall-clock data obtained for each package. Future work will include instrumented data as part of the performance analysis as access to these tools becomes available on the new systems.

The ARL MSRC's 256 processor I686 Linux Cluster (Powell), the 2,048 processor EM64T Linux Cluster (JVN), and the 32-core dual-core Intel Woodcrest test-bed Linux Cluster (Ping) were used to perform benchmark runs all codes in the suite. OVERFLOW and GAMESS were run with configurations of 1, 2, 4, 8, 16, and 32 processors. CFD++ was run on 8, 16, 24, and 32 processing element (PEs) due to memory limitations presented by the large simulation being run. CTH was run with 1, 2, 4, 8, and 16 PEs due to a problem running with 32 PEs on Ping. The overall wall-clock run time for each processor configuration was recorded for each benchmark.

2. Benchmark Code Suite

CTH: This code is used for modeling multidimensional, multimaterial, large deformation, strong shock wave propagation problems in solid mechanics. It uses advanced numerical methods coupled with advanced material models to model the non-linear behavior of materials subjected to large deformations under high strain rate loading[1]. The executable used for this benchmark was built with the PGI compiler on all platforms. Interprocessor communication was accomplished through the MPICH Message Passing Interface (MPI) libraries on Powell and JVN and the OpenMPI MPI library on Ping. The input used for this benchmark is a simulation of a projectile impacting a ten inch concrete slab at 470m/s with a stop time of 0.012 seconds.

CFD++: This code is a general-purpose computational fluid dynamics (CFD) code for accurate and efficient flow simulations. Its unified-grid, unified-physics and unified-computing methodology applies to all flow regimes, all types of mesh and topologies[1]. The

0-7695-3088-5/07 $25.00 © 2007 IEEE

executable used for this benchmark was built with the Intel compiler on all platforms. Interprocessor communication was accomplished through the MPICH MPI libraries on Powell and JVN and the OpenMPI MPI library on Ping. The input used for these benchmarks is for a finned projectile at Mach number 2.5 at zero angle-of-attack utilizing approximately 2.25 million cells. The calculation is run for 250 time steps.

GAMESS: GAMESS (General Atomic and Molecular Electronic Structure System) is a program for *ab initio* quantum chemistry. It can compute wave functions ranging from RHF, ROHF, UHF, GVB, and MCSCF, with CI and MP2 energy corrections for these. Analytic gradients are available for these self-contained field functions, for automatic geometry optimization, transition state searches, and reaction path following [1]. The executable used for this benchmark was built with the Intel compiler on all platforms. Also, all interprocessor communication is accomplished through sockets rather than through MPI library calls as is used in the other benchmark codes. The input used for these benchmarks utilizes the popular method of the second order energy correction of Moller-Plosset perturbation theory (MP2) with the molecule benzoquinone. Benzoquinone is a neutral singlet molecule with 24 atoms and 108 electrons. An energy gradient calculation was performed on Benzoquinone. The basis set employed is Pople's double-zeta basis[2] with an additional 'd' polarization function on the heavy atoms (6-31G*).

OVERFLOW: This code is based on Overset structured grids (Chimera). Geometry complexity is reduced to a set of relatively simple overlapping body-fitted grids and topologically simple background grids. The structure of the individual grid components facilitates viscous boundary layer resolution, implicit time-integration algorithms, and efficient use of computer memory[1]. The executable used for this benchmark was the double-precision version built with the PGI compiler on all platforms. Interprocessor communication was accomplished through the MPICH MPI libraries on Powell and JVN and the OpenMPI MPI library on Ping. The input case used for this benchmark is a 5.56 M193 bullet for the M16 rifle series, consisting of 2.6 million grid points with an initial speed of Mach 2.5 and angle-of-attack of 2 degrees.

3. Hardware/Software Configuration

Table 1 provides the hardware configuration for the ARL Linux Clusters used for this study.

Table 1. System Hardware Configurations

System	Processor Type	Processor Speed	Number of Processors	Processors/Cores per Node	Memory per Node	Communication Architecture	Communication Speed
Powell	IA-32	3.06 GHz	256	2	2GB	Myrinet	2 GB/sec
JVN	EM64T	3.6 GHz	2048	2	4GB	Myrinet	2 GB/sec
Ping	Intel dual-core Woodcrest	3.0 Ghz	32	4	8 GB	Infiniband	10GB/sec

The following operating system and application support software were available for building and running the application codes on Powell and JVN: Intel, PGI, and GNU compilers (C, C++, FORTRAN 77/90) and the MPICH/Myrinet message-passing software library. The following operating system and application support software were available for building and running the application codes on Ping: Intel, PGI, and GNU compilers (C, C++, FORTRAN 77/90) and the OpenMPI/Infinband or MPICH/Myrinet message-passing software library. The Sub Grid Engine batch scheduler was used on Powell and the load sharing facility batch scheduler on JVN and Ping.

4. Application Performance Results and Analysis

Figures 1 through 8 plots the timings obtained for each of the benchmarks on each platform, as well as, the scalability of each of the codes. Below is a discussion of each of the benchmarked codes.

4.1. CTH

Figures 1 and 2 depict the comparative performance and scalability respectively of CTH, as run on 1, 2, 4, 8, and 16 processors on Powell, JVN, and Ping. The results

clearly show the performance improvement of the Woodcrest technology in Ping over the EM64T and IA-32 architectures of JVN and Powell respectively. Ping runs were from 3.3 to 4 times faster than those on Powell and ranged from 1.5 to 2.2 times faster than the same runs on JVN. Since there was no instrumentation software available for MJM it is difficult to pinpoint the exact cause for this improvement. However, the following system enhancements on MJM are the most likely contributors: larger memory per node on MJM (8GB) then Powell (2GB) and JVN (4GB) and level 2 cache of Powell (8MB vs. 1MB) and JVN (8MB vs. 2MB), the much faster communication speeds of the Infiniband interconnect on Ping verses the Myrinet interconnect on Powell and JVN, and the fact that MJM has 4 cores per node verses 2 per node on the other two systems. The scalability, with the maximum being 5.6 on Powell, proved to be disappointing. We believe this was mainly due to the small data size of the input.

Figure 1. CTH Performance

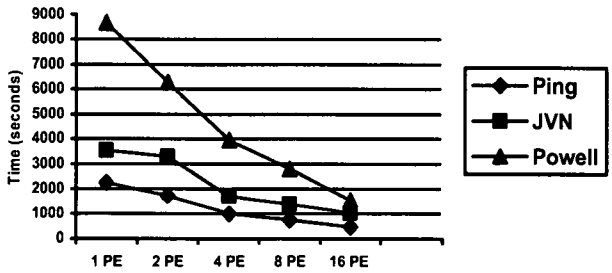
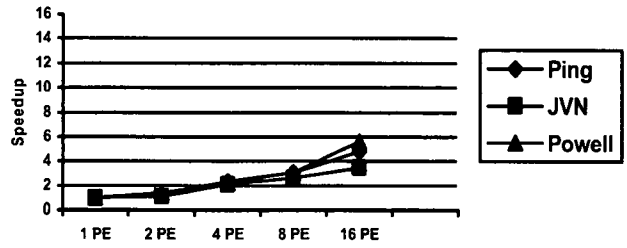


Figure 2. CTH Scalability



4.2. CFD++

Figures 3 and 4 depict the comparative performance and scalability respectively of CFD++, as run on 8, 16, 24, and 32 processors on Powell, JVN, and Ping. These benchmarks once again clearly demonstrate the improvement in performance that each subsequent architectural generation achieved over the previous generation. For CFD++, however, the Woodcrest/ Infiniband technology seemed to provide a significant advantage over the previous technologies. Its performance was twice that of Powell and 1.6 to 1.9 times faster than JVN. It also exhibited very good scalability with almost a three times speed up on four times as many processors.

Figure 3. CFD++ Performance

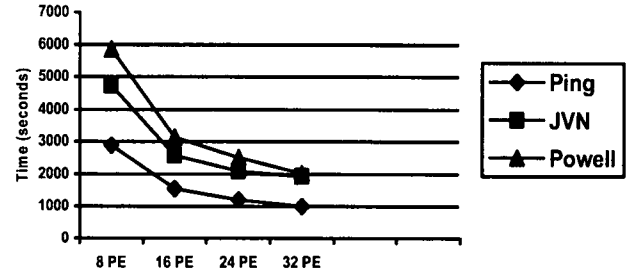
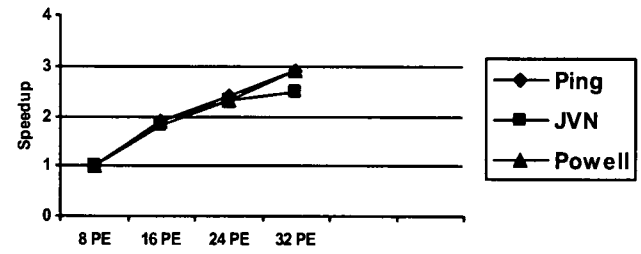


Figure 4. CFD++ Scalability



4.3. GAMESS

Figures 5 and 6 depict the comparative performance and scalability respectively of GAMESS, as run on 1, 2, 4, 8, 16, and 32 processors on Powell, JVN, and Ping. Although all machines demonstrate the longest execution times for the least amount of CPUs and decrease in time as the number of CPUs increase, JVN shows a minimum at 2 CPUs. This can be attributed to JVN's nodes being composed of two processors per node and the need to use Myrinet to communicate to the other nodes. The older, 32-bit Powell system also has two processors per node and utilizes Myrinet, but strangely doesn't exhibit this behavior. The newer Ping system utilizes the much faster Infiniband and clearly does not show this behavior. Clearly, Ping has the fastest compute times. For one processor, the Woodcrest chips are about four times faster than the corresponding single processor job on JVN. These differences in multiples increase as the number of CPUs increase until at 32 processors, Ping is more than 12 times faster than JVN and Powell. Interestingly, the execution time of Powell is about the same as JVN. Ping

in not only faster than JVN or Powell, but its architecture allows a much better speedup than either JVN or Powell as Ping's speedup is much closer to the theoretical speedup line. Also, it appears Powell's architecture is more efficient than JVN's as Powell's speedup curve is closer to the theoretical curve than JVN's.

Figure 5. GAMESS Performance

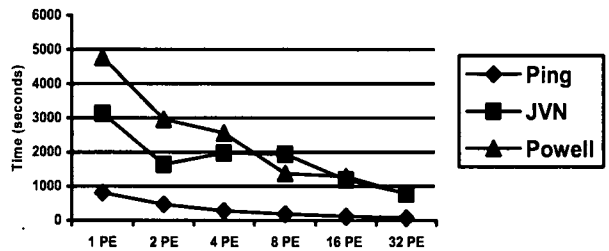
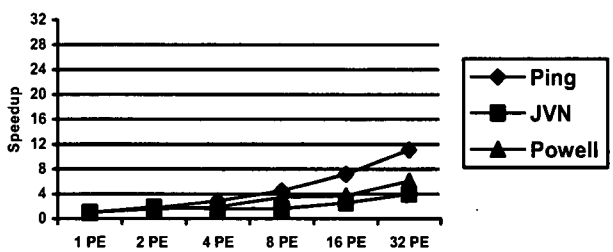


Figure 6. GAMESS Scalability



4.4. OVERFLOW

Figures 7 and 8 depict the comparative performance and scalability respectively of OVERFLOW, as run on 1, 2, 4, 8, 16, and 32 processors on Powell, JVN and Ping. Ping was shown to perform from 3.1 to 4.8 times faster than Powell. The main contributor to this performance difference is probably the use of the double precision version of OVERFLOW. Since Powell is a 32-bit architecture, all double precision operations must be simulated in software rather than performed within the hardware as is done on JVN and Ping. JVN, however, is only slightly slower than Ping, except in the 4PE case, where it slightly outperformed Ping. Since the Infiniband communication architecture is five times faster than the Myrinet communication architecture this was somewhat unexpected. There appears to be a twofold explanation to this anomaly. The first reason is that the code was designed well in terms of mitigating the communication overhead through the use of non-blocking communication. The second explanation appears to be that the Woodcrest chip is exhibiting a similar behavior to previous clusters in that using fewer than the maximum number of processors per node (cores in this case) leads to significantly better performance of parallel jobs. In this case, rerunning the 4 PE case using 2 nodes with 2 MPI tasks per node, improved the run time from 1960 to 1380 seconds. In terms of scalability, Ping demonstrated noticeably better scalability at all processor counts between 2 and 16, but all three architectures demonstrated a speedup of approximately 16.7 on 32 processors.

Figure 7. OVERFLOW Performance

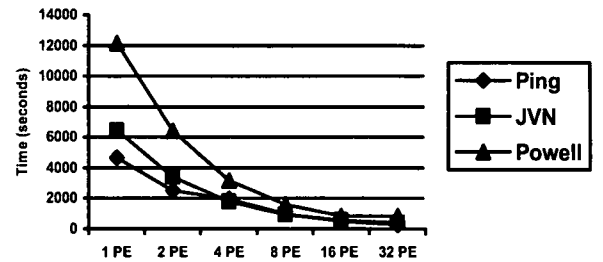
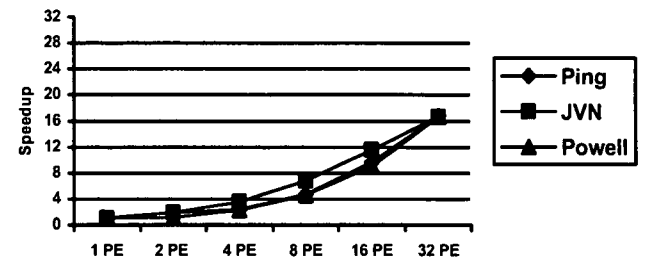


Figure 8. OVERFLOW Scalability



5. Conclusions

The results of all four benchmarks clearly demonstrate the increased capability of each generation of the large-scale clusters that have been introduced into the ARL compute environment. It is also evident that some codes have benefited more than others, but all have benefited. And, certainly, access to instrumentation software on the new Woodcrest system would have allowed a more detailed and definitive discussion as to the source of the performance improvements across the systems. We hope to be able to perform that analysis as soon as the instrumentation software is available and present the analysis results at a later date. Combining the results of this study with the fact that each succeeding generation also provides significantly more processors per system it is evident that computational capabilities of the ARL MSRC have been greatly enhanced through the acquisition of these systems.

Acknowledgements

This work was supported in part by a grant of computer time from the DoD High Performance Computing Modernization Program at the ARL MSRC.

References

1. HPC Modernization Program software listings, www.hpcmo.hpc.mil/CHSSI/software.html.
2. Ditchfield, R., W.J. Hehre, and J.A. Pople, *J. Chem. Phys.*, 54, 724, 1971; V.A. Rassolov, M.A. Ratner, J.A. Pople, P.C. Redfern, and L.A. Curtiss, *J. Comp. Chem.*, 22, 976, 2001.